Discrete Applied Mathematics 20 (1988) 181–190
North-Holland

MAXIMAL CHORDAL SUBGRAPHS

P.M. DEARING, D.R. SHIER and D.D. WARNER
Department of Mathematical Sciences, Clemson University, Clemson, SC 29634, USA

Received 20 November 1984
Revised 17 February 1987

An algorithm for finding maximal chordal subgraphs is developed that has worst-case time complexity of $O(|E|\Delta)$, where $|E|$ is the number of edges in G and Δ is the maximum vertex degree in G. The study of maximal chordal subgraphs is motivated by their usefulness as computationally efficient structures with which to approximate a general graph. Two examples are given that illustrate potential applications of maximal chordal subgraphs. One provides an alternative formulation to the maximum independent set problem on a graph. The other involves a novel splitting scheme for solving large sparse systems of linear equations.

1. Introduction

Let $G=(V,E)$ be a finite, undirected, loopless graph without multiple edges, where V is the vertex set and E is the edge set. Denote an edge of E by $\{u,v\}$ where $u,v\in V$. A graph G is *chordal* (triangulated, rigid circuit) if it does not contain any cycle of length greater than three as an induced subgraph [10]. For a given graph $G=(V,E)$ we consider the problem of finding a spanning subgraph $G'=(V,E')$ of G, where E' is a maximal subset of E such that (V,E') is chordal. Alternatively, one can view the problem as determining a minimal set of edges to delete from E so that the resulting graph is chordal.

The motivation for considering this problem stems from a general approach of approximating arbitrary graphs by subgraphs (or supergraphs) having special structure. Chordal subgraphs provide one instance of this approach. There are frequently graph optimization problems for which efficient algorithms (and data structures) exist for chordal graphs, but which are NP-hard for general graphs. Thus given a difficult instance of such an optimization problem, exact computations on chordal subgraphs can yield useful approximations to the solution for the given graph. Two well-known problems in discrete mathematics are examined in which chordal approximations lead to new solution approaches. Further evidence of the usefulness of chordal subgraphs is provided in a recent paper by Balas and Yu [2]. They present an algorithm, with worst-case time complexity $O(|V|+|E|)$, for finding a maximal *induced* chordal subgraph, which they then use in a branch and bound algorithm for finding maximum cliques in a general graph. A recent paper by Balas [1] shows how (edge) maximal chordal subgraphs can be used in solving the maximum weight clique problem.

In Section 3 we present an algorithm for finding a maximal chordal subgraph of an arbitrary graph G, and in Section 4 we show that this algorithm can be implemented with a worst-case time complexity bound of $O(|E|\Delta)$ where Δ is the maximum degree of any vertex in G. In Section 5, the maximum independent set problem is formulated using maximal chordal subgraphs. This formulation is stronger than the customary edge-vertex formulation and leads to a Lagrangian relaxation approach. In Section 6, maximal chordal subgraphs are used to define a splitting for the coefficient matrix of a large sparse system of linear equations. This splitting is then used to produce iterative schemes for solving the original system of equations. Because of chordality, these iterative schemes can be implemented so that no additional storage is required beyond that needed to store the original system.

2. Background

The relationship between chordal graphs and Gaussian elimination in solving systems of linear equations is well documented [13, 14]. We briefly review that relationship in order to introduce notation and motivate the consideration of maximal chordal subgraphs.

For a graph $G=(V,E)$, with $n=|V|$, an *ordering* α of V is a bijection of $\{1,2,\ldots,n\}$ onto V. For each v in V, the *neighborhood* of v is the set $N(v)=\{u\in V: \{u,v\}\in E\}$. A vertex v is *simplicial* in G if $N(v)$ is complete. It is well known [6, 9, 10] that every chordal graph has a simplicial vertex (in fact at least two) and any induced subgraph of a chordal graph is likewise chordal. Thus if a simplicial vertex and its incident edges are removed from a chordal graph, the resulting chordal subgraph has a simplicial vertex. For a chordal graph, let α be defined by the order in which simplicial vertices are successively removed in this process. Then α is called a *perfect elimination ordering* (peo). Alternatively, given an ordering α of V, define the *monotone adjacency set* of v as

$$M(v)=\{u\in N(v): \alpha^{-1}(v)<\alpha^{-1}(u)\}.$$

Then α is a peo if and only if $M(v)$ is complete for all $v\in V$. Rose [13, 14] and Dirac [6] have shown that a graph is chordal if and only if it possesses a peo. Algorithms for finding perfect elimination orderings in chordal graphs have been given by Fulkerson and Gross [9], Rose, Tarjan and Leuker [15], Shier [17], and Tarjan and Yannakakis [18].

For a subset S of V, let $E(S)=\{\{x,y\}\in E: x\in S,\ y\in S\}$ and let $G(S)=(S,E(S))$ denote the subgraph of G induced by S. For any v in V, define the *deficiency* of v by

$$D(v)=\{\{x,y\}: \{x,v\}\in E,\ \{y,v\}\in E \text{ and } \{x,y\}\notin E\}.$$

The *vertex elimination* operation for vertex v removes v and its incident edges from G and adds the edges in $D(v)$, yielding the graph $(V-\{v\}, E(V-\{v\})\cup D(v))$.

For some ordering α, suppose this vertex elimination operation is repeatedly applied to the graph $G=(V,E)$, using the vertices $\alpha(1),\ldots,\alpha(n)$ in turn. Then the resulting set of deficiencies is the *fill-in*, denoted by

$$F(\alpha)=\bigcup_{i=1}^{n} D(\alpha(i)).$$

The *elimination graph* is defined by $G_\alpha=(V,E\cup F(\alpha))$. Clearly α is a peo of the chordal graph G_α. Of course, if α is a peo of G, then $F(\alpha)=\emptyset$ and $G_\alpha=G$. Chordal graphs are particularly attractive for certain applications because they suffer no fill-in.

Let $A=[a_{ij}]$ be an $n\times n$ matrix with symmetric nonzero structure. A graph $G=(V,E)$ can be associated with A by taking $V=\{1,2,\ldots,n\}$ and $E=\{\{i,j\}: a_{ij}\neq 0,\ 1\leq i<j\leq n\}$. It is well known [13] that if a_{kk} is chosen as a pivot element for Gaussian elimination, the new nonzero elements created by the pivot operation correspond to the deficiency of vertex k, and if pivots are performed according to the ordering α of V, the resulting set of nonzero elements created corresponds to the fill-in $F(\alpha)$ of G. Thus if G is chordal and α is a peo, no fill-in will result if the indices of the pivot elements are chosen as $\alpha(1),\ldots,\alpha(n)$.

Attempts to preserve sparsity in large systems of equations have led to a search for orderings that generate 'small' fill-in. Rose, Tarjan and Lueker [15] describe an algorithm for finding an ordering α with minimal fill-in; this algorithm has a complexity bound of $O(|V||E|)$. Yannakakis [21] has shown however that the problem of finding an ordering with minimum fill-in is NP-complete.

If α is an ordering which gives minimal fill-in, then one may think of the graph G_α as a minimal chordal supergraph of G. That is, $F(\alpha)$ represents a minimal set of edges that can be added to E to make G chordal. We now turn this problem around and ask instead for a *maximal chordal subgraph*. Namely, we seek a minimal set of edges that can be deleted from E to make G chordal. Because chordality is not preserved under graph complementation, finding maximal chordal subgraphs is separate from and is not reducible to finding minimal chordal supergraphs.

The computational complexity of finding a maximum chordal subgraph is still an open question. Yannakakis [20] has demonstrated the NP-completeness of several related problems, in which a minimum set of edges is to be deleted to obtain a subgraph with a certain property.

3. A maximal chordal subgraph algorithm

The algorithm described here for finding a maximal chordal subgraph of an arbitrary graph builds upon the maximum cardinality search (MCS) algorithm [18] for finding a peo of a chordal graph. The MCS algorithm constructs a peo α in reverse order $\alpha(n),\alpha(n-1),\ldots,\alpha(1)$. At stage k, $S_k=\{v\in V: \alpha^{-1}(v)\geq k\}$ denotes the set of vertices ordered thus far. A vertex v_0 in $V-S_k$ is chosen so that $N(v_0)\cap S_k$ has

maximum cardinality, and the algorithm sets $\alpha(k-1)=v_0$. Because vertices v_0 are chosen in this fashion, it follows that $N(v)\cap S_k$ is complete for all $v\in V-S_k$. Indeed, since a peo is actually constructed by this procedure [18], $M(v)$ is complete and so then is $N(v)\cap S_k\subseteq M(v)$.

The algorithm proposed for finding a maximal chordal subgraph of an arbitrary graph $G=(V,E)$ likewise constructs sets S_k by adjoining one vertex at each stage. A subset of edges E', with $E'\subseteq E$, is generated by adjoining edges to E' at each stage so that chordality is always maintained. For each $v\in V-S_k$, we maintain a set $C(v)$ of vertices in S_k that are adjacent to v and form a complete set with respect to the edges in E'. A crucial step is to choose a vertex v_0, whose set $C(v_0)$ has largest cardinality, to enter S_k at each stage. The sets $C(v)$ and E' are then updated appropriately.

The algorithm below is claimed to produce a maximal chordal subgraph $G'=(V,E')$ of any graph $G=(V,E)$.

Algorithm MAXCHORD

1. For all $v\in V$ set $C(v):=\emptyset$.
   Define $k:=|V|$, $E':=\emptyset$.
   Select any $v_0\in V$, set $S_k:=\{v_0\}$ and $\alpha(k):=v_0$.

2. For all $u\in V-S_k$ with $\{u,v_0\}\in E$
   if $C(u)\subseteq C(v_0)$ then
   $C(u):=C(u)\cup\{v_0\}$,
   $E':=E'\cup\{u,v_0\}$.

3. Select $v_0\in V-S_k$ such that
   $|C(v_0)|\geq|C(v)|$ for all $v\in V-S_k$.

4. Set $\alpha(k-1):=v_0$, $S_{k-1}:=S_k\cup\{v_0\}$, $k:=k-1$.
   If $k>1$ then go to Step 2, else STOP.

We now prove that MAXCHORD produces a maximal chordal subgraph of the given graph $G=(V,E)$.

Lemma 1. *If $G=(V,E)$ is chordal, then* MAXCHORD *produces $G'=G$, and* MAXCHORD *and* MCS *produce the same* peo *for G.*

Proof. We use induction on $n=|V|$. Clearly the conclusions hold for $n<2$. Assume the lemma is true for all graphs with $k<n$ vertices.

Let $G=(V,E)$ be a chordal graph with $|V|=n$, and let α be the peo produced by MCS. Suppose MAXCHORD produces the subgraph $G'=(V,E')$ with $E'\subseteq E$. We show that MAXCHORD also produces the peo α and that $E\subseteq E'$, whence $G=G'$.

Let H be the subgraph of G induced by $V-\alpha(1)$. Then H is chordal and $\alpha(n),\ldots,\alpha(2)$ is a peo for H produced by MCS. By the inductive assumption,

MAXCHORD also produces the peo $\alpha(n),\ldots,\alpha(2)$ for H and MAXCHORD produces the graph H, so $E'(H)=E(H)$.

Under the peo α produced by MCS for G, the vertex $x=\alpha(1)$ is a simplicial vertex of G. Let $N(x)=\{v_1,\ldots,v_r\}$ and suppose the v_i are ordered so that $\alpha^{-1}(v_1)>\cdots>\alpha^{-1}(v_r)$. Note that $E=E(H)\cup\{\{x,v_1\},\ldots,\{x,v_r\}\}$.

By the inductive assumption, MCS and MAXCHORD both choose v_1 to enter S_k at the same stage. At this stage $C(x)=\emptyset$, so $C(x)\subseteq C(v_1)$, and the edge $\{x,v_1\}$ is added to E'. When v_2 is chosen to enter S_k, $C(x)=\{v_1\}$. Since $\{v_1,v_2\}\in E'(H)$ we have that $v_1\in C(v_2)$. Thus $C(x)\subseteq C(v_2)$ and edge $\{x,v_2\}$ is added to E'. Continuing, we see that all edges $\{x,v_i\}$, $i=1,\ldots,r$ are added to E'. Thus $E\subseteq E'$, so $G'=G$. Also the peo α is produced by MAXCHORD. □

Lemma 2 [15]. *If $G_1=(V,E_1)$ and $G_2=(V,E_2)$ are both chordal with $E_1\subset E_2$, then there is some edge $e\in E_2-E_1$ such that $G_2-e=(V,E_2-\{e\})$ is chordal.*

Lemma 3. *If $G_1=(V,E_1)$ and $G_2=(V,E_2)$ are both chordal with $E_1\subset E_2$, then there is some edge $e\in E_2-E_1$ such that $G_1+e=(V,E_1\cup\{e\})$ is chordal.*

Proof. Lemma 3 follows by successively applying Lemma 2. □

Theorem 1. MAXCHORD *produces a maximal chordal subgraph $G'=(V,E')$ of $G=(V,E)$.*

Proof. Let α be the ordering produced by MAXCHORD when applied to $G=(V,E)$. At each stage k and for each vertex v in $V-S_k$, $C(v)$ is a complete set of vertices in $G'(S_k)$. Thus at each stage k the vertex $\alpha(k-1)$ is a simplicial vertex of the subgraph of G' induced by S_{k-1}. Consequently the ordering α is a peo of G' and G' is chordal.

To show that G' is a maximal chordal subgraph, it suffices by Lemma 3 to show that for any edge $e\in E-E'$, the graph $G'+e$ is not chordal. Suppose $G'+e$ is chordal. Let $e=\{p,q\}$ and suppose $k=\alpha^{-1}(p)>\alpha^{-1}(q)$. Then the induced subgraphs $[G'+e](S_k)$ and $G'(S_k)$ are identical, are both chordal, and have as their edge sets $E'(S_k)$. By applying MAXCHORD to $G(S_k)$, the ordering $\alpha(n),\alpha(n-1),\ldots,\alpha(k)$ produced is a peo for $[G'+e](S_k)=G'(S_k)$. When vertex p is added to form S_k, edge $\{p,q\}$ is considered for addition to E'. We remark that edge $\{p,q\}$ can only be included at this step. When processing G, edge $\{p,q\}$ was not included in E', so $C(q)$ cannot be a subset of $C(p)$. However, when processing $G'+e$, MAXCHORD must produce $G'+e$ by Lemma 1, so that $\{p,q\}$ is included in the edge set, i.e., $C(q)\subseteq C(p)$. This contradiction implies that $G'+e$ is not chordal. □

Fig. 1 shows an example of a graph and a maximal chordal subgraph generated by MAXCHORD. The algorithm was initialized at node a and deleted edges are

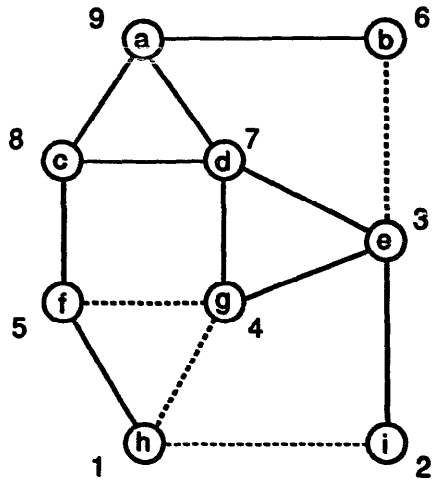

Fig. 1. A maximal chordal subgraph generated by MAXCHORD.

indicated by dashed lines. The numbers adjacent to the vertices correspond to the perfect elimination ordering produced by MAXCHORD.

A careful examination of the proofs given in Lemma 1 and Theorem 1 reveals that a more general result has in fact been established. Namely, suppose that we construct the subgraph $G'=(V,E')$ of G by iteratively applying a *vertex selection step* (VSS) and then an *edge inclusion step* (EIS). Then such an algorithm will also be guaranteed to produce a maximal chordal subgraph G' under the following conditions. First, the VSS should be based exclusively on edges currently in E' and it should produce a peo with $G'=G$ when run on a chordal graph G. (For example, selecting vertices according to the lexicographically largest label [10] relative to the current subgraph G' would be a valid choice.) Second, the EIS should ensure that the neighborhoods $N(v)\cap S_k$ remain complete for all $v\in V-S_k$.

4. Implementation

Algorithm MAXCHORD can be implemented to run in $O(|E|\Delta)$ time, where Δ is the maximum degree of any vertex in G. It is assumed that the graph is represented in forward star format [10, p. 37] in which the vertices adjacent to a given vertex occupy consecutive storage locations. There are two main data structures used in this implementation. First, each complete set $C(x)$ is maintained as a singly-linked list, where vertex $y\in C(x)$ is represented by $\alpha^{-1}(y)$ and the linked list $\alpha^{-1}(y_1)\rightarrow \alpha^{-1}(y_2)\rightarrow\cdots\rightarrow\alpha^{-1}(y_r)$ is kept in increasing order: $\alpha^{-1}(y_1)<\alpha^{-1}(y_2)<\cdots<\alpha^{-1}(y_r)$. Second, each vertex $u\in V-S_k$ is associated with the jth element of an array, where $j=|C(u)|$. Vertices u having the same $|C(u)|$ are chained together using a doubly-linked list. This structure enables an element v_0 of maximum cardinality to be found quickly, via an address calculation sort [4].

In Step 2 of MAXCHORD, checking for $C(u)\subseteq C(v_0)$ can be carried out in at

most $O(\Delta)$ time by a modified merge sort, since the linked lists for $C(u)$ and $C(v_0)$ are ordered. This step must be carried out for each edge in E. Hence the time complexity of the entire implementation (which is dominated by Step 2) is at most $O(|E|\Delta)$. Moreover, space requirements are linear, $O(|V|+|E|)$, for this implementation.

5. An application to discrete optimization

The maximum independent set problem provides an application of maximal chordal subgraphs in a discrete optimization setting. Given a graph $G=(V,E)$ with $|V|=n$, a set of vertices $I \subseteq V$ is an *independent set* if no two vertices in I are adjacent in G. A *maximum independent set* has maximum cardinality over all independent sets in G.

The maximum independent set problem on a graph is an important problem occurring in discrete optimization. It arises as a natural formulation for many combinatorial problems, and a wide class of discrete optimization problems can be transformed into maximum independent set problems. The problem has been studied extensively [12] and is known to be NP-hard.

The following formulation of the problem exploits the structure of chordal subgraphs. Let $G'=(V,E')$ be a maximal chordal subgraph of G. Recall that a *clique* of G' is a maximal complete subgraph of G'. Let C be the clique-vertex incidence matrix of G' and let N be the edge-vertex incidence matrix for all edges in $E-E'$. (Note that C is easily obtained from the output of MAXCHORD.) The maximum independent set problem can then be written as the following

$$\begin{aligned} P\text{:}\quad & \max ex \\ & \text{s.t. } Cx \le e, \\ & \quad Nx \le e, \\ & \quad x = 0,1 \end{aligned}$$

where x is an n-vector, each component x_i corresponds to a vertex v_i in V, and e is a vector of ones with appropriate dimension.

Problem P has some advantages over the edge-vertex formulation, say P', of Nemhauser and Trotter [12], in which each constraint corresponds to an edge of the graph G. Let R and R' be the linear programming relaxations of problems P and P', respectively: i.e., the constraint $x=0,1$ is replaced by $x \ge 0$. Let $v(Q)$ denote the optimal objective function value of a problem Q. Then problem R provides an upper bound for problem P that is at least as good as R': namely, $v(P) \le v(R) \le v(R')$.

Another advantage of problem P is that it has a useful Lagrangian relaxation [7, 16], which for any nonnegative vector u with dimension $|E-E'|$ is given by

$$\begin{aligned} LR\text{:}\quad & \max ex + u(e-Nx) = z(u) \\ & \text{s.t. } Cx \le e, \\ & \quad x = 0,1. \end{aligned}$$

Problem LR may be interpreted as finding a maximum weighted independent set on the chordal subgraph G'. The vertex weights are given by the components of the vector $(e-uN)$. It is well known [7, 16] that $z(u)$, for any $u \geq 0$, provides an upper bound on the objective function value of P; moreover, for the Lagrangian dual

$$D: \quad \min\{z(u),\ u \geq 0\} = z(u^*),$$

we have $z(u^*) = v(R) \geq v(P)$.

Efficient solution procedures exist for problem LR for any $u \geq 0$. Since chordal graphs are perfect [10], the linear programming relaxation of LR with $x = 0, 1$ replaced by $x \geq 0$ produces integer optimal solutions. Alternatively, Frank [8] gives a linear-time algorithm for solving LR based on a perfect elimination ordering of G' (which is generated by MAXCHORD).

These observations suggest an enumeration scheme using problem LR to compute upper bounds. At each node of the branching tree, an upper bound is obtained by computing an approximate solution to D using a subgradient approach to iterate on the vector u; see [7, 16].

6. An application to sparse systems of linear equations

We consider the problem of solving the system of linear equations $Ax = b$ where A is large and sparse. Let A be an $n \times n$ matrix, not necessarily symmetric, with diagonal matrix $D = \text{diag}(a_{11}, a_{22}, \ldots, a_{nn})$. Let $G = (V, E)$ be the graph associated with the zero-nonzero structure of $A - D$, and let $G' = (V, E')$ be a maximal chordal subgraph of G. We define C to be the negative of the matrix associated with G': that is, the nonzeros of C are given by $c_{ij} = -a_{ij}$ for $\{i, j\} \in E'$. Also N denotes the *remainder matrix* defined by the expression $A = D - C - N$. The decomposition $A = D - C - N$ is termed a *maximal chordal splitting* of the matrix A, where $D - C$ is the corresponding *maximal chordal submatrix* of A.

A maximal chordal splitting can be used as the basis for a number of iteration schemes to solve $Ax = b$. For example, the simplest iteration is based on the linear system

$$(D - C)x^{k+1} = Nx^k + b. \tag{1}$$

Since $D - C$ corresponds to a chordal graph we can find a perfect elimination ordering so that (1) can be solved without introducing fill-in. This implies that by using maximal chordal subgraphs we can build iteration schemes for solving $Ax = b$ which will never require more storage than that necessary to store A in the first place.

Two practical questions immediately arise. First, what conditions guarantee that the linear system (1) can be solved stably using the perfect elimination ordering? Second, what conditions guarantee that the iteration scheme, based on a maximal chordal splitting, will converge? One natural case to consider is when A is an M-matrix. Recall that an $n \times n$ matrix A is a *nonsingular M-matrix* if $A = sI - B$ with

$s>0$, $B\geq 0$ and $s>\varrho(B)$, where $\varrho(B)$ is the spectral radius of B.

If A is a nonsingular M-matrix, then the matrices C and N will be nonnegative and $\varrho(sI-D+C)\leq\varrho(sI-D+C+N)$. Consequently the matrix $D-C$ is also a nonsingular M-matrix. This has two important consequences. First, $D-C$ is diagonally similar to a strict diagonally dominant matrix [3] and hence Gaussian elimination with a perfect elimination ordering is a stable algorithm for solving the linear system in (1). Second, since $D-C$ is an M-matrix and N is a nonnegative matrix, the splitting is 'regular' and the iteration defined by (1) converges [19]. Thus maximal chordal splittings yield an interesting class of storage-efficient iterative methods for M-matrices. Extensions to more general matrices are possible using techniques analogous to the shifted incomplete Cholesky factorization [11].

If a nonsingular M-matrix is also symmetric, then it is a *Stieltjes* matrix. It is a nontrivial fact that Stieltjes matrices are positive definite [3]. Since the maximal chordal splitting preserves symmetry, $D-C$ is a Stieltjes matrix whenever A is. Thus such a splitting can be used with the generalized conjugate gradient algorithm of Concus, Golub and O'Leary [5]. An interesting feature of their algorithm is that in infinite precision arithmetic the algorithm will terminate in a finite number of steps. With a maximal chordal splitting, the number of steps equals the rank of the remainder matrix N. Combining the extraction of maximal chordal subgraphs with such a scheme may then prove to be computationally attractive.

References

[1] E. Balas, A fast algorithm for finding an edge-maximal subgraph with a TR-formative coloring, Discrete Appl. Math. 15 (1986) 123–124.

[2] E. Balas and C.S. Yu, Finding a maximum clique in an arbitrary graph, SIAM J. Comput. 15 (1986) 1054–1068.

[3] A. Berman and R.J. Plemmons, Nonnegative Matrices in the Mathematical Sciences (Academic Press, New York, 1979).

[4] A.T. Berztiss, Data Structures: Theory and Practice (Academic Press, New York, 1975).

[5] P. Concus, G.H. Golub, and D.P. O'Leary, A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations, in: J.R. Bunch and D.J. Rose, eds., Sparse Matrix Computations (Academic Press, New York, 1976) 309–332.

[6] G.A. Dirac, On rigid circuit graphs, Abh. Math. Sem. Univ. Hamburg 25 (1967) 71–76.

[7] M.L. Fisher, Lagrangian relaxation methods for solving integer programming problems, Management Sci. 27 (1981) 1–18.

[8] A. Frank, Some polynomial algorithms for certain graphs and hypergraphs, Proc. 5th British Combinatorial Conference (1975) 211–226.

[9] D.R. Fulkerson and O.A. Gross, Incidence matrices and interval graphs, Pacific J. Math. 15 (1965) 835–855.

[10] M.C. Golumbic, Algorithmic Graph Theory and Perfect Graphs (Academic Press, New York, 1980).

[11] T.A. Manteuffel, Shifted incomplete Cholesky factorization, Sparse Matrix Proceedings 1978 (SIAM, Philadelphia, 1979) 41–61.

[12] G.L. Nemhauser and L.E. Trotter, Vertex packings: structural properties and algorithms, Math. Programming 8 (1975) 232-248.

[13] D.J. Rose, Triangulated graphs and the elimination process, J. Math. Anal. Appl. 32 (1970) 597-609.

[14] D.J. Rose, A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations, in: R. Read, ed., Graph Theory and Computing (Academic Press, New York, 1973) 183-217.

[15] D.J. Rose, R.E. Tarjan and G.S. Lueker, Algorithmic aspects of vertex elimination on graphs, SIAM J. Comput. 5 (1976) 266-283.

[16] J.F. Shapiro, A survey of Lagrangian techniques for discrete optimization, Annals of Discrete Math. 5 (1979) 113-138.

[17] D.R. Shier, Some aspects of perfect elimination orderings in chordal graphs, Discrete Appl. Math. 7 (1984) 325-331.

[18] R.E. Tarjan and M. Yannakakis, Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs, SIAM J. Comput. 13 (1984) 566-579.

[19] R.S. Varga, Matrix Iterative Analysis (Prentice-Hall, Englewood Cliffs, NJ, 1962).

[20] M. Yannakakis, Edge-deletion problems, SIAM J. Comput. 10 (1981) 297-309.

[21] M. Yannakakis, Computing the minimum fill-in is NP-complete, SIAM J. Algebraic Discrete Methods 2 (1981) 77-79.